Distributed Network Management Security

February xx, 1997

Authors:

Paul Meyer
Secure Computing Corporation (SCC)
Paul_Meyer@securecomputing.com

1. Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months. Internet-Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as a ''working draft'' or ''work in progress.''

To learn the current status of any Internet-Draft, please check the 1id-abstracts.txt listing contained in the Internet-Drafts Shadow Directories on ds.internic.net, nic.nordu.net, venera.isi.edu, or munnari.oz.au.

2. Abstract

Use of SNMP to securely manage distributed networks through firewalls has not been formally described, although features critical to such management are included in SNMP. This document reports on a study performed at Secure Computing Corporation on a method to solve this management function. The project name this study occurred under is Distributed Network Management Security.

Slight modifications to the SNMP V2 User-Based Security Model (RFC 1910) and a conceptual redeployment of some of the functions contained within this model provide a basis for this study. The acronym DNMS will be used in this document to refer to the modifications.

The basis for the DNMS extensions is a firewall platform that contains at least two distinct network stack implementations, one for the exterior, or public network, and one for the interior, or protected network. DNMS consists of two SNMP V2 proxies, one on each network, with the security-related functions implemented in a third component that also serves as the communication path between the two proxy components.

This implementation allows the management and use of SNMP security to be concentrated in the firewalls, where it is assumed that the threats being protected against lie outside the firewall, somewhere out on the public network.

3. Protocol Modifications

The primary modification to the SNMPv2usec [12,13] header is to provide semantics on the userName field. This modification would allow the use of constructs such as the X.500 Distinguished Name in the userName, which in turn would allow the use of a certificate infrastructure as an adjunct to the SNMPv2usec model for key distribution. The form of the userName was taken from a draft of the PKIX working group.[14]

SNMPv2usec defines the header in terms of an OCTET STRING rather than using the SNMPv2 SMI[5]. A direct copy of the GeneralName ASN.1 construct was not used due to the confusion a standard BER-encoded field could cause. Instead, the field was defined as follows:

<userLen> a one octet value containing the length of the <userName>
<userType> an octet value indicating which of the GeneralName choices the <userName> consists of.
<userName> the string representation of the userName.

DNMS uses (experimentally) a <model> value of 99 to indicate the different header.

The GeneralName construct is defined as the following:

```
GeneralName  ::=  CHOICE  {
     otherName        [0] INSTANCE OF OTHER-NAME,
     rfc822Name       [1] IA5String,
     dNSName          [2] IA5String,
     x400Address      [3] ORAddress,
     directoryName    [4] Name,
     ediPartyName     [5] IA5String,
     url              [6] IA5String }
```

Use of GeneralName choice 0 (otherName) indicates a standard SNMPv2usec <userName>.

The final modification is to potentially allow either SNMPv2 [5-11] PDUs or SNMPv1 [1,2] PDUs to be contained inside the protected data field. This was intended for proxy applications. Rules for translating between SNMPv1 and SNMPv2 are occasionally ambiguous and other people are looking at this particular problem. SCC was intent on investigating the feasibility of the SNMPv2 proxy concept in firewalls and in using a certificate infrastructure to manage the authentication and encryption portions of the problem.

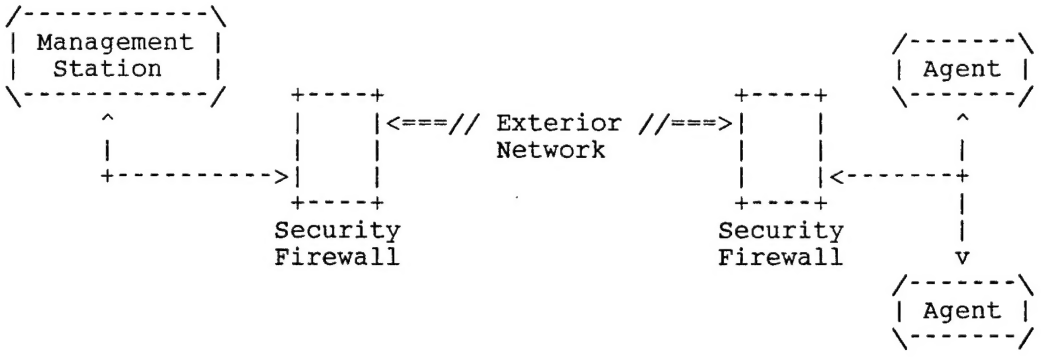
For completeness, the resultant header is shown below.

```
Message ::=
        SEQUENCE{
                version
                        INTEGER { v2 (2) },
                parameters
                        OCTET STRING,
                -- <model=99>
                --      <qoS><agentID><agentBoots><agentTime><maxSize>
                --      <userLen><userType><userName><authLen><authDigest>
                --      <contextSelector>
                data
                        CHOICE {
                                plaintext
                                        PDUs,
                                encrypted
                                        OCTET STRING
                        }
        }
```

It should be noted that a very similar transform could be made to the SNMPv2* proposal. SCC chose SNMPv2usec as the base as there were no implementations of SNMPv2* readily available for experimentation.

4. High Level Usage Model

The idea behind the model is to concentrate the security processing of SNMP PDUs in a pair of firewalls. A management station can reside behind one of the firewalls. By using the modified protocol described in this document between the firewalls, it can communicate with devices behind a firewall at a distant location without the management station itself undergoing modification. This is shown below.



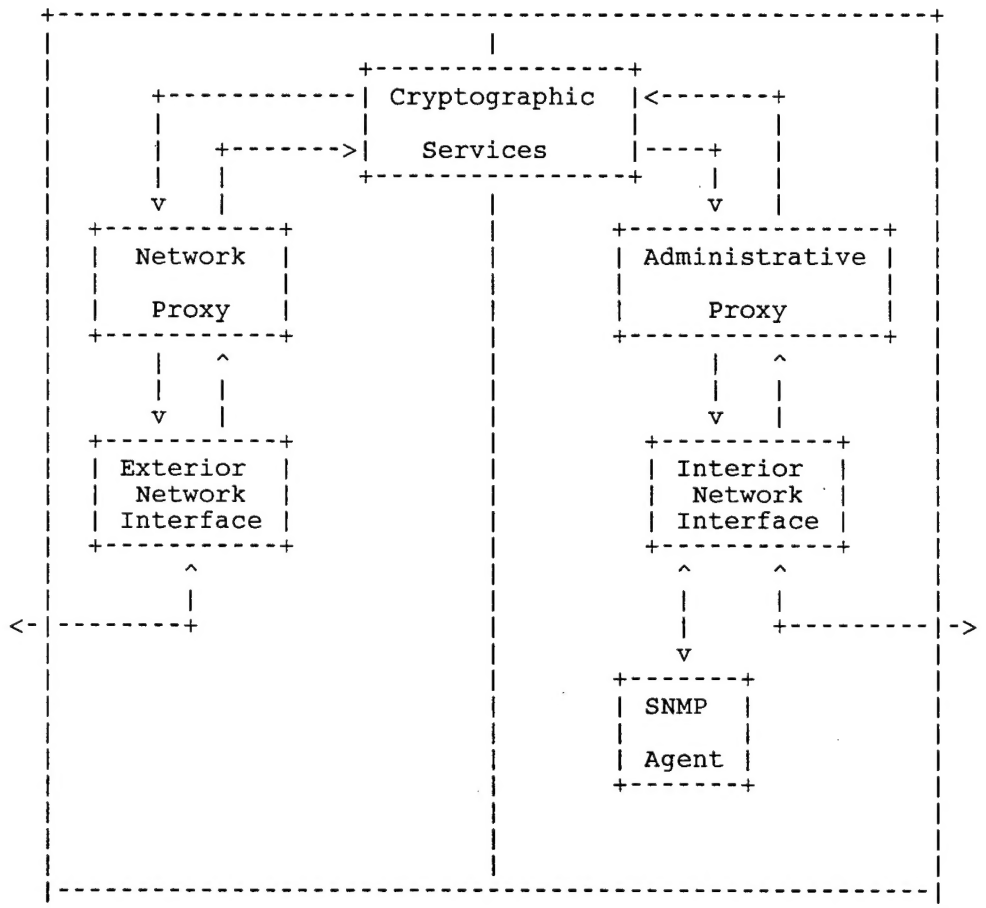
The firewall implementation requires a system that supports two logically distinct network stacks. A simple SNMP proxy is attached to the exterior network. This proxy is set up to accept either DNMS or SNMPv2usec messages. DNMS messages indicating a local-proxy type in the <contextSelector> are passed on to the crypto process for handling through the firewall. Messages received by this proxy from the crypto process are sent on the exterior network as DNMS messages. SNMPv2usec or DNMS messages that contain a local context are processed with MIB support for the System group ONLY. This proxy is referred to as the network proxy.

A more sophisticated proxy is attached to the interior network. This proxy requires that the network interface pass all SNMP messages NOT directly addressed to the firewall to the proxy (this is usually a feature of application-level firewalls). Based upon the administration information in the PDU received from the interior network, this proxy selects a <contextSelector>, <qoS>, <agentID>, and possibly <userName> for transmission to the other firewall. This mapping is contained in the Local Configuration Datastore (LCD). A DNMS header is then put on the SNMP PDU in place of the previous header, and the message is passed to a cryptographic services handler. Messages received from this crypto handler (which were received by the network proxy on the exterior network) have the reverse transformation applied to them, with either a SNMPv2usec or a SNMPv1/SNMPv2c [4] message being sent out on the interior network. This proxy is referred to as the administrative, or admin proxy.

A third module performs as a communications interface between these two proxies, and also handles all cryptographic functions. Keys for authentication and encryption are obtained from the LCD, which contains locally cached certificates (userTypes other than 0) or the configured key information. For cases when the certificate is not

present, the crypto process will attempt to obtain the certificate via other means. These could be X.500 Directory requests, DNSSEC lookups, or some other method.

A fourth module embodies the SNMP agent for the firewall, including the MIBs that contain the admin proxy LCD information and the cryptographic information for the crypto process. This agent is directly accessed on the interior network from either management stations or via the administrative proxy.



The DNMS model does not comply with one of the central assumptions made with SNMP, that being all central functions are self-contained within SNMP. Here DNMS expressly allows the option of using external means for key management, such as an X.500 Directory or DNSSEC.

5. Current Status

A prototype implementation of DNMS is currently under development. This prototype is based upon the CMU SNMPv2usec beta distribution, with extensive modifications to recognise contexts and pass proxy information to the crypto processing. The crypto processing is based upon a generic proxy toolkit, with extensions for the certificate cache and the programmatic DUA.

This prototype implementation has passed most of the basic message types with varying degrees of <qoS> set. "Interesting" misconfigurations have been also tested (usually accidentally).

The MIBs supporting the administrative proxy's LCD and the crypto process' LCD are to be defined, along with the supporting code. Communication between the admin proxy and the firewall agent has been minimally tested; support of this MIB should fully test the interface.

6. Processing Details

6.1 Interior Network through Admin Proxy

Accept the message from the transport layer as usual and begin validation according to the normal rules.

SNMPv2usec messages are minimally transformed.
The user field is verified against configured users in the LCD. If none exists (i.e. no secrets for this user), the message cannot be transported if the configured <qoS> indicates authentication or encryption are required. Return Authorization Error in this case (as usual for SNMPv2, a Report is generated if the <qoS> indicates).

Verify that the <contextSelector> is local-proxy. In the LCD, find the matching localContext entry, and pick up the remoteContext that matches. If there is no match, return Authorization Error. Also, the configuration of the firewall should allow no local type contexts to the admin proxy; they should have been passed to the SNMP agent. Return General Error in this case.
Rebuild the usec header into a DNMS header. The <userName> is set up as a type 0. <agentID> is set to the receiving DNMS, <agentBoots> and <agentTime are unmodified>. <contextSelector> is set to the remoteContext from the LCD.

SNMPv2c and SNMPv1 messages undergo the same processing. The LCD is searched for the community found in the header. If none is found, treat it as a bad community. If found, pull up the matching remoteContext, <qoS> and <userName> from the LCD, along with the last indicators for <agentBoots> and <agentTime> (the details for <agentBoots> and <agentTime> have not been completed at this time). Use the values to build a DNMS header.

The message is then passed to the crypto process for authentication.

6.2 Outbound Cryptograhic Processing

Crypto processing will first validate that the message is a DNMS message (<model> = 99). If not, a General Error is returned to the admin proxy, which may or may not be able to return a Report PDU to the originator.

If the <qoS> indicates report PDUs are valid, then the request ID (from the PDU), <contextID>, and a configurable time window are stored. This provides additional replay protection above and beyond <agentTime> handling in the admin proxy.

For report/response type PDUs, this stored time is checked against the current time to see if the message can be processed. Failure generates the NotInWindows authentication error.

If the <qoS> indicates no authentication or encryption required, the message is passed on to the network proxy without modification.

When the <qos> specifies authentication, the <userName> is used to query for keying material. The crypto process will first search the LCD for a match, then will check a local certificate cache. Only when both these have failed will the crypto proxy look to an exterior certificate infrastructure (such as an X.500 directory) for keying material.

Authentication is then performed as in RFC1910 to build the <authDigest>.

If confidentiality protection is also set in the <qoS>, the crypto process will obtain the privacy keying material. To avoid padding, confidentiality is provided by using DES in Cipher Feedback (CFB) mode. (If this proves too slow, a padding scheme will be implemented and either Electronic Codebook (ECB) or Cipher Block Chaining (CBC) mode will be used.) The data portion is then encrypted.

6.3 Network Proxy to Exterior Network

The message is received from the crypto proxy and the <contextSelector> is picked up. Based upon the <contextSelector>, select the network address of the receiving DNMS. Send the message via the exterior network stack.

(We have used the convention of building the <contextSelector> out of the IP address that handles the context, a unique tag, and a corresponding value that would map to a view name. This reduces the number of configurable items - always a good thing in experimental work.)

6.4 Exterior Network through Network Proxy

Accept the message from the transport layer as usual and begin validation according to the normal rules.

SNMPv1, SNMPv2c, and SNMPv2usec messages are accepted and handled with a minimal agent that supports the system group only. This should give an indication to a management station that the system needs to communicate with these protocol extensions to validly forward messages through the proxy.

DNMS messages should contain a local-proxy <contextSelector>. If not, they are failed with Authorization Error. In addition, the <contextSelector> and <userName> must exist in the LCD to prevent the Authorization Error.

Valid messages are passed to the crypto process.

6.5 Inbound Cryptographic Processing

Messages received from the network proxy are passed through to the admin proxy if the <qoS> indicates that no confidentiality or authentication was applied to the message. As with outbound crypto processing, the request-ID and the times are checked to verify that the message is in the time window before being passed through.

As with the outbound proxy, the <userName> is used to find keying materials in the same order (LCD, local cache, remote methods). If no keying material is found, an Authentication Error is generated. The message is then decrypted (if necessary) and the <authDigest> validated. An Authentication Error is generated if these fail.

Pending the time window checks, the message is then passed through to the admin proxy.

6.6 Admin Proxy to Interior Network

Use the passed <contextSelector> to find the transform in the LCD.
For a DNMS to SNMPv2usec message, pick up the matching localContext, downgrade the <qoS> as configured, and rebuild the header to SNMPv2usec format (the <agentID> for the end entity is also in the LCD). Send the message.
For a SNMPv2c or SNMPv1 message, pick the matching community string for the passed <contextSelector>. Save the latest indications for <agentBoots> and <agentTime> for use on responses/next requests. Rebuild the header in community format and send the message to the end entity.
Unknown contexts at this point are silently discarded. When the model extensions discussed in section 5 occur, an Inform will be used to signal the peer DNMS that this occurred.

7. Future Work

7.1 Impact of the SNMP Advisory Team's Work

Members of the DNMS team were unable to attend the December 1996 IETF. Examination of the materials available as of December 26th indicate that much of the DNMS crypto processing is contained within modules defined in the handouts; these could be easily extended to handle the GeneralName construct and certificate processing as performed in the DNMS crypto process. The admin proxy transforms would be placed in the proxy application; support of things like SNMPv1 and SNMPv2c PDUs would be dependent upon individual implementations.

7.2 Additional Protocol Modifications

Since the start of the SNMP security programs, network level encryption has been becoming more widespread. Use of network level encryption could be especially important when communicating between two firewalls as in the DNMS model. It would mitigate the traffic analysis threat where the SNMP header is in cleartext. To utilize a

network level encryption capability, an additional <qoS> would be added that would indicate authorization required, use network level privacy if supported, else use SNMP privacy. This would allow DNMS messages to pass through a new class of firewall/routers that forward messages with valid IPSEC/IPv6 associations in a very efficient path.

7.3 Use of Informs Between Firewalls with AutoDiscovery

The transformation rules require that a unique context be shared by the two firewalls for for each unique manager to agent's context pairing. This is similar to the party problem with SNMPv2classic, where configuration information critical for communication was either required before SNMP would work or scaled poorly.

The SNMP support of the interior network could be determined by use of a number of the autodiscovery algorithms that have come into play, such as limited ping sweeps and ARP snooping followed by simple SNMP requests. Informs could then be used to share information between the DNMS firewalls and with the management station to build knowledge of the network.

8. Experience So Far

DNMS so far has simply moved the configuration complexity from the configuration of security secrets to the proxy context infrastructure. The distman working group's advances in the autodiscovery area is believed to ease this concern. It would also make things easier if the MIB for the proxy context translations existed.

The second concern is in the correctness of the <agentBoots> and <agentTime> handling. A secondary time window was added to guard against replays; a good analysis of a working model needs to be performed to see whether or not the handling can be improved and whether or not this secondary time window is necessary.

9. Security Concerns

Security concerns are the basis for this memo.

10. References

[1] RFC 1155, Case, J.D., et. al., Structure and Identification of Management Information for TCP/IP-based Internets. (SNMPv1), May 1990

[2] RFC 1157, Case, J.D., et. al., Simple Network Management Protocol (SNMPv1), May 1990

[3] RFC 1158, Rose, M.T., Management Information Base for Network Management of TCP/IP-based Internets: MIB-II May 1990

[4] RFC 1901, Case, J.D., et. al., Introduction to Community-based SNMPv2. (SNMPv2c), Jan. 1996

[5] RFC 1902, Case, J.D., et. al., Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2), Jan. 1996

[6] RFC 1903, Case, J.D., et. al., Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2), Jan. 1996

[7] RFC 1904, Case, J.D., et. al., Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2), Jan. 1996

[8] RFC 1905, Case, J.D., et. al., Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPV2), Jan. 1996

[9] RFC 1906, Case, J.D., et. al., Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2), Jan. 1996

[10] RFC 1907, Case, J.D., et. al., Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2), Jan. 1996

[11] RFC 1908, Case, J.D., et. al., Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework, Jan. 1996

[12] RFC 1909, McCloghrie, K., An Administrative Interface for SNMPv2, Feb. 1996

[13] RFC 1910, Waters, G., User-based Security Model for SNMPv2 (SNMPv2u), Feb. 1996

[14] Housley, R., et. al., Internet Public Key Infrastructure, Part I: X.509 Certificate and CRL Profile, Feb. 1996 (work in progress)